AFRL-VA-WP-TR-2001-3026

DEVELOPMENT OF A FLEXIBLE OPTIMIZATION CAPABILITY

GARRET N. VANDERPLAATS
DIPANKAR K. GHOSH
JOHN H. GARCELON
VLADIMIR BALABANOV
GERHARD VENTER

VANDERPLAATS RESEARCH AND DEVELOPMENT, INC.
1767 S. 8TH STREET, SUITE 100
COLORADO SPRINGS, CO 80906

NOVEMBER 1999

FINAL REPORT FOR PERIOD 14 APRIL 1997 – 30 SEPTEMBER 1999

THIS IS A SMALL BUSINESS INNOVATION RESEARCH (SBIR) PHASE 2 REPORT

AIR VEHICLES DIRECTORATE
AIR FORCE RESEARCH LABORATORY
AIR FORCE MATERIEL COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OH 45433-7542

NOTICE

USING GOVERNMENT DRAWINGS, SPECIFICATIONS, OR OTHER DATA INCLUDED IN THIS DOCUMENT FOR ANY PURPOSE OTHER THAN GOVERNMENT PROCUREMENT DOES NOT IN ANY WAY OBLIGATE THE UNITED STATES GOVERNMENT. THE FACT THAT THE GOVERNMENT FORMULATED OR SUPPLIED THE DRAWINGS, SPECIFICATIONS, OR OTHER DATA DOES NOT LICENSE THE HOLDER OR ANY OTHER PERSON OR CORPORATION; OR CONVEY ANY RIGHTS OR PERMISSION TO MANUFACTURE, USE, OR SELL ANY PATENTED INVENTION THAT MAY BE RELATED TO THEM.

THIS REPORT IS RELEASEABLE TO THE NATIONAL TECHNICAL INFORMATION SERVICE (NTIS). AT NTIS, IT WILL BE AVAILABLE TO THE GENERAL PUBLIC, INCLUDING FOREIGN NATIONS.

THIS TECHNICAL REPORT HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION.

Victoria A. Tischler
VICTORIA A. TISCHLER
Aerospace Engineer
Structural Design and Development Branch

Terry M. Harris
TERRY M. HARRIS, Chief
Structural Design and Development Branch
Structures Division

CHRISTOPHER L. CLAY, Chief
Structures Division
Air Vehicles Directorate

Do not return copies of this report unless contractual obligations or notice on a specific document require its return

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
November 1999	FINAL	04/14/97 – 09/30/99

4. TITLE AND SUBTITLE
Development of a Flexible Optimization Capability

5a. CONTRACT NUMBER
F33615-97-C-3204

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
65502F

6. AUTHOR(S)
Garret N. Vanderplaats
Dipankar K. Ghosh
John H. Garcelon
Vladimir Balabanov
Gerhard Venter

5d. PROJECT NUMBER
3005

5e. TASK NUMBER
41

5f. WORK UNIT NUMBER
9C

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Vanderplaats Research and Development, Inc.
1767 S. 8th Street, Suite 100
Colorado Springs, CO 80906

8. PERFORMING ORGANIZATION REPORT NUMBER
VRD 0001

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Vehicles Directorate
Air Force Research Laboratory
Air Force Material Command
Wright Patterson, AFB OH 45433-7542

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/VASD

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
AFRL-VA-WP-TR-2001-3026

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for Public Release; Distribution Unlimited

13. SUPPLEMENTARY NOTES
THIS IS A SMALL BUSINESS INNOVATION RESEARCH (SBIR) PHASE 2 REPORT

14. ABSTRACT
This report describes the activities performed during the SBIR Phase II contract entitled, "Development of a Flexible Optimization Capability". The objective was twofold. The first was to create a graphical user interface to couple optimization and related design capabilities with third party analysis software. The second objective was to create a general design study tool which can effectively utilize the information available from the optimization and related processes.
A graphical user interface was created to couple optimization with third party analysis software such as, ABAQUS and LS-DYNA. An object relational database system, a gradient based optimization functional module, a design of experiments (DOE) functional module and a post processing module, were developed during the contract period. All these developments, except the databasesystem, have been combined into a software system called VisualDoc. Version 1.0 of VisualDoc was released in December 1998, and Version 1.2 was released in July 1999. VisualDOC 2.0, scheduled to be released in the first quarter of 2000, will contain the database system and a JAVA based graphical user interface.

15. SUBJECT TERMS
Optimization, Automated Design, CAE, Multidisiciplinary Design Optimization, Design Study Tools, Design of Experiments (DOE), Structural Optimization, Mathematical Programming

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Victoria A. Tischler
Unclassified	Unclassified	Unclassified	SAR	46	19b. TELEPHONE NUMBER *(Include area code)* 937-255-9729

Table of Contents

List of Figures

FOREWORD

This report is submitted in fulfillment of the requirements of a Small Business Innovative Research (SBIR) Phase II, Contract No. F33615-97-C-3204, entitled "Development of a Flexible Optimization Capability." This report covers the performance activities from April 14, 1997 to October 14, 1999. Two separate documents, namely VisualDOC Version 1.0 Reference Manual, and VisualDOC 1.2 Supplement have been separately submitted to the AFRL/Wright-Patterson to fulfill the requirements of this contract.

The work was performed at Vanderplaats Research & Development (VR&D), Inc. No subcontractor was involved. This work was the second phase of a continuing two-phase SBIR contract funded by AFRL/Wright-Patterson. The contributors to this contract are: Dr. Garret N. Vanderplaats (PI), Dr. Dipankar K Ghosh, Dr. John H. Garcelon, Dr. Vladimir Balabanov and Dr. Gerhard Venter.

At AFRL/Wright-Patterson, Ms Victoria A. Tischler was the project engineer and Dr. V. B. Venkayya was the initiator of the contract. The technical advice and assistance received from Dr. Venakyya, Ms tischler and others during the course of this contract performance are gratefully acknowledged.

For additional information as to acquiring or using the VisualDOC software, please contact VR&D at 719-473-4611, Ext. 104 or email to sales@vrand.com. Additional information on VisualDOC and other VR&D products are also available at VR&D's web page at http://www.vrand.com.

Introduction

The objective of this SBIR Phase II research effort was two fold. The first was to create a graphical user interface to couple optimization and related design capabilities with third party analysis software. The second objective was to create a general design study tool which can effectively utilize the information available from the optimization and related processes.

Such a design study tool, called VisualDOC, has been developed during the course of this study. This report provides a brief summary of the research and development activities that were performed during the contract period. This report should be read in conjunction with two other reports called VisualDOC 1.0 Reference Manual and VisualDOC 1.2 Supplement Manual, which have been submitted to AFRL/Wright-Patterson.

Section 1 of this report briefly discusses the purpose of the research efforts and outlines a plan of action that was undertaken.

Section 2 provides a detailed outline of research & development activities that were performed. Different modules of the design study system called VisualDOC are presented and discussed.

Section 3 discusses the continuing R&D activities related to this present research efforts.

Section 4 briefly provides an overview of the related R&D works that were considered complimentary to the present work.

Section 5 provides a list publications related to this SBIR project that were published in technical conferences and meetings.

Section 6 provides some early success stories of using VisualDOC software by industrial clients.

Section 7 presents a list of references directly related to this research project.

1.0 Background

In Phase I of this research effort, we addressed the issue, why optimization technology did not receive the degree of acceptance in the engineering community that might be expected based on the power of the technology. Two key reasons were identified as contributing to this lack of use. First was the cost and risk of adding sophisticated technology to existing commercial analysis software when a clear market has not been demonstrated. Second was the recognition of the fact that optimization is generally considered to be quite specialized, requiring special expertise to use

These two issues were addressed by concentrating our research efforts on four major areas:

- Addition of a graphical user interface (GUI) to existing VR&D optimization software DOC/DOT to demonstrate the coupling of nonlinear structural analysis software with optimization.
- Identify new concepts that can be added to optimization technology to make it more robust for engineering design.
- Identify ways to present the optimization process and results in a more understandable fashion and define ways to perform "what if" studies as a post-processing operation.
- Define an overall software program structure which is flexible and expandable and which can provide a blueprint for Phase II development and eventual commercialization.

The key objective of Phase II was to convert these research ideas and plans developed in Phase I into a viable commercial product that will encourage expanded applications of design optimization technology.

Thus, the objective was to create new computer software that will provide the engineer with a general and efficient design tool for optimization studies. This will help expand the use of optimization in a design study environment. While nonlinear constrained optimization will provide a core capability, the goal here is much more ambitious. Specifically, we want to create extensive graphical capabilities to create the tasks related to design optimization, visualize the optimization process, retain important information for later use, and provide a range of post-processing capabilities for further study and understanding of the design optimization process.

Although we did not plan to create Multidisciplinary Design Optimization (MDO) capabilities directly, it was VR&D's goal to create a very general design study capability for a *multitude* of disciplines as the mechanism for gaining acceptance of this technology. By creating a general design framework, the extension to fully integrated MDO can follow quickly as users recognize the need.

These objectives were achieved by addressing the following specific issues:

Overall Program Architecture

The general program structure defined in Phase I was expanded in detail. This defined the inter-relations between the graphical interface, the computational modules, the database, and third party software.

Program Database

An object relational database was created for storage of analysis and design information, optimization process history and related information. The general concept of object relational database was chosen for its flexibility and consistency with object oriented programming. This was considered to provide the most efficient means of interfacing with a variety of data which would be encountered using multiple third party software packages.

Graphical User Interface

In Phase I, a basic graphical user interface (GUI) was created for the Windows operating system. This was further refined and extended to other computing platforms, including all major UNIX workstations. This was considered to be a key issue in ease of use and eventual acceptance of the final software. Using objected-oriented programming, the relationships between the GUI, database, computational modules and third party software were formalized. Because this is what the user sees, particular emphasis was placed on clear pre-processing, optimization progress tracking and post-processing.

Functional Modules

Each of the functional modules included in Version I of the final product was designed in detail. The DOT optimizer, with appropriate enhancements, was chosen to be the nonlinear programming module. Design of experiments (DOE) and associated response surface modules were developed and included in Version 1.0 of the final product.

Interfaces to Third Party Analysis Software

Two nonlinear structural analysis programs, namely ABAQUS and LS-DYNA were interfaced with initial release of the software. We also created an interface to MATLAB's computational engine. Using this interface, analysis programs written in MATLAB's M-files could be directly called by the design optimization software.

Also, interfaces to CAD software have been created. The initial candidate was MSC/PATRAN from MSC.Software. Preliminary forms to create design information have been completed.

Documentation

Documentation was an ongoing task throughout Phase II. All software components contain extensive internal documentation as well as written technical documentation. User documentation was also created early as a guide to programming, as well as later becoming a user's manual and reference guide. User documentation was created for both on-line and hard copy formats. The online documentation was developed in both Adobe PDF and HTML formats.

In September 1998, we released the initial version of the new software called VisualDOC. Version 1.0 of VisualDOC was released without the design database. This release did not contain all the components of the research efforts that we have been developing during the last two years. The strategy was to get the software out to our existing customers so that we can receive their feedback to further improve the software. All existing clients received VisulaDOC and had an opportunity to comment on it. This was an interim

release and provided us an opportunity to test the market, and gather user input for the next release of VisualDOC. A number of users, both from the industry and research institutions, are presently either using or evaluating VisualDOC.

In July 1999, we released VisualDOC Version 1.2. The following additional capabilities were included:

- DOE functional module and mechanisms to transfer DOE results to the formal VisualDOC optimization.
- VisualDOC stand alone version; this will eliminate the need of a compiler to work with the VisualDOC software system.
- New VisualDOC online help system in the form of both compiled HTML and stand alone HTML.
- JAVA based post-processing module mainly for VisualDOC on UNIX platforms.

The marketing of the VisualDOC program started before the end of Phase II through promotion of this software to VR&D's current client base. This included providing development versions of the software for beta testing by selected clients. Once VisualDOC was formally released, it was provided to a number of users for evaluation. Our strategy was to work closely with existing clients to further enhance and refine the software developed here.

2.0 Research & Development

2.1 Approach

The goal here was to create the most sophisticated design optimization software available anywhere. To achieve this, we proposed to create software that allows the user to

efficiently perform design studies in a unified way. Thus, while optimization was the key ingredient, a variety of complimentary tools have been developed which operate in concert with optimization to assist the engineer in a thorough investigation of design options and alternatives.

It was recognized at the outset that many existing commercial analysis programs do not contain optimization and those that do are limited in their ability to easily perform the trade-off studies essential to creating a high quality product. Thus, there is a need for a general framework where the engineer can perform optimization using the analysis software of his or her choice, including programs that contain their own optimization capability, while retaining or creating additional information needed for rapid "what if" studies.

To achieve this goal, the work plan was divided into the following seven parts, which are discussed in detail below:

- Overall Program Architecture
- Graphical User Interface
- Object Relational Database System
- Functional Modules
- Interfaces to Third Party Analysis Software
- Post Processing
- License Manager

2.2 Overall Program Architecture

The basic program architecture was developed during Phase I of this project. During Phase II, this was the first to be refined and expanded. Here, a very detailed flowchart of the program was created, with particular emphasis on expandability. This is shown in Figure 1.

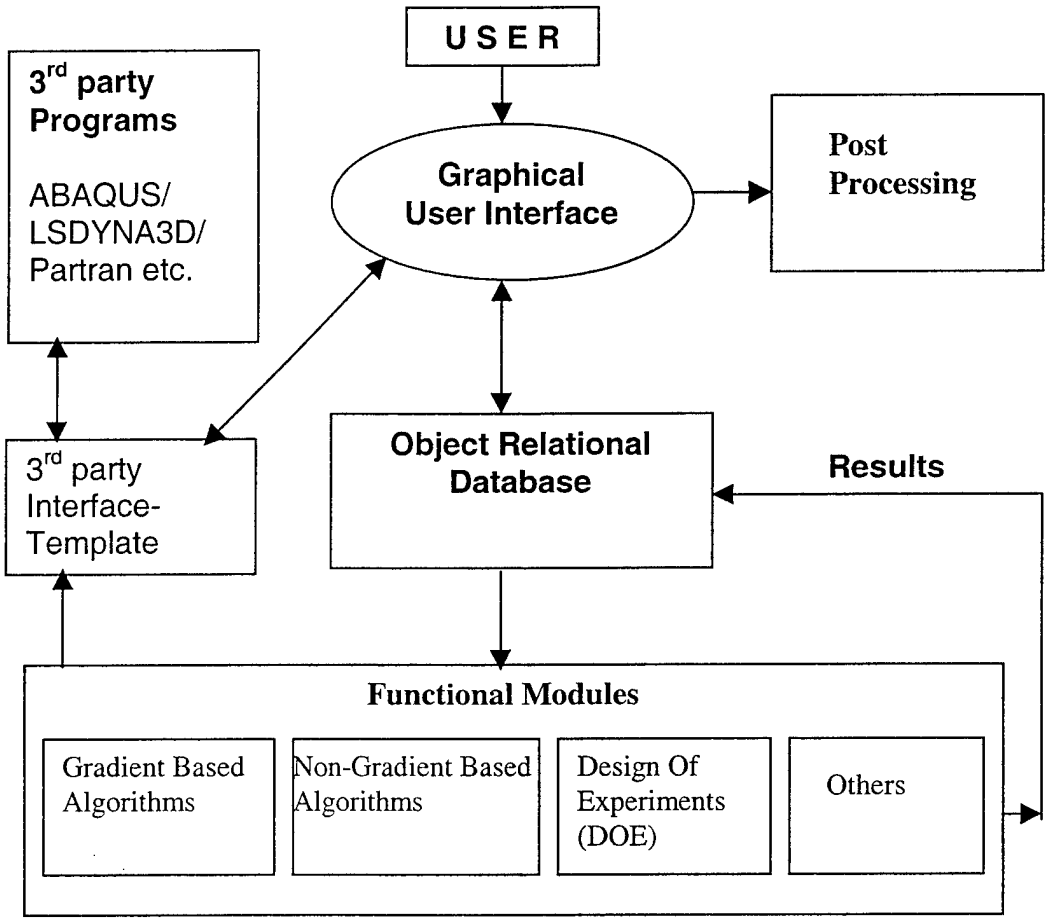

Figure 1: Overall Program Architecture

The user defines the design data using the graphical user interface (GUI). The GUI functions in both the pre- and post-processing phases of the design study. At the pre-processing stage, the design model is defined. For example, the user will specify the

design variables, objective function(s), and constraints, if any. The user will also specify which functional module to use and any necessary control parameters. At the post-processing stage, the user may specify which results to retrieve from the database and graphically display. The user will be able to perform "what-if" studies about the design during post-processing.

The GUI stores all design data in the database using an object-relational format. The specified functional module reads design data from the database, and performs the specified task. The design results are then transferred back to the same database for further processing. The further processing of design results may consist of viewing the results graphically, restarting the design optimization, modifying the design parameters etc.

Even though the emphasis of the current effort was to provide a convenient capability to perform optimization using a single analysis module, it is clear that the user will quickly realize that multiple disciplines are involved in real design and their interactions cannot be avoided. Therefore, an important part of this effort was to provide the flexibility to include multiple disciplines in the near term and to formally include several disciplines in an overall study in the future. This expectation was a primary reason why the effort to create a formal data management system was a principal part of the proposed effort.

Here we define an overall program architecture that will allow the engineer to perform single discipline design studies while accommodating multidisciplinary optimization in the future. Indeed, by having the database available, several disciplines can perform design studies using consistent data immediately.

In the following sub-sections, we will provide more details of the different components of the program architecture.

2.3 Graphical User Interface

During the Phase I, a basic graphical user interface (GUI) to VR&D's optimization program DOC/DOT (later it became a part of the VisualDOC optimizer) was written. This was written on a Windows' platform using XVT software and the C language. During the Phase II, this GUI went through several enhancements and refinements.

For windows platforms, it was redesigned using the Microsoft Foundation Class (MFC) and the C++ programming language. For UNIX platforms, it was re-written using Motif library and the C programming language.

The Windows 95/NT version of the GUI was redesigned using object-oriented techniques. Internally, the program creates objects for each design entity, such as objectives, design variables, constraints, etc. Externally, the user interacts with the program using a spreadsheet metaphor. The user may open and manipulate multiple views of the design data, while the program maintains data consistency through a layer of objects that provide an object interface between the views of the data and the internal structure (Figure 2).

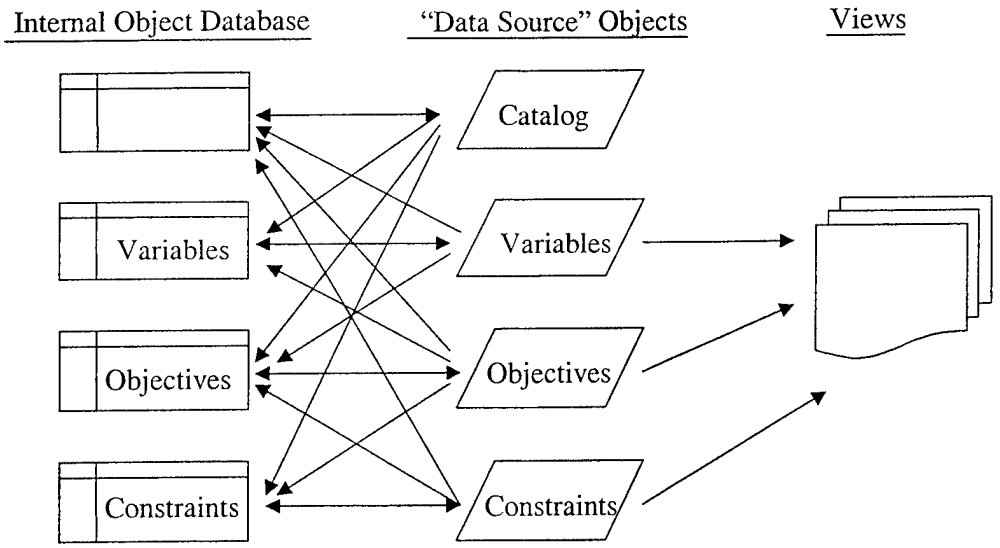

Figure 2: Interaction between Data Objects and Views

Using an object-oriented design, the program transfers data between the internal database of design objects and the user's views of that data via "data source" objects. This allows the program to maintain data consistency at two levels. The first level occurs when the user enters data into a spreadsheet view. Before the program commits this data to the internal database, the data source objects perform basic consistency checks. If the data passes the consistency rules, the internal database is updated. The internal database subsequently signals any other spreadsheet views that a database change has occurred. This signal causes the other spreadsheet views to check for new or updated information.

The second level of error checking occurs when transferring design data to the VisualDOC optimizer. It is critical at this point to verify the data that will be presented to VisualDOC optimizer since all required information must be instantiated and valid. This provides a high level of confidence that the data provided to the optimizer will run.

The UNIX version of the GUI was created using the Motif library and the C language. Functionally the UNIX version is similar to that of the Windows version except for the look-and-feel graphical components. In the windows version, the design information is entered using spreadsheets whereas in the UNIX version, the design data is entered using table formats.

Using the GUI, the user can enter design data for the optimizer, and view results generated by the optimizer. Figures 3 and 4 show the VisualDOC organization on the Windows 9x/NT and UNIX platforms, respectively.

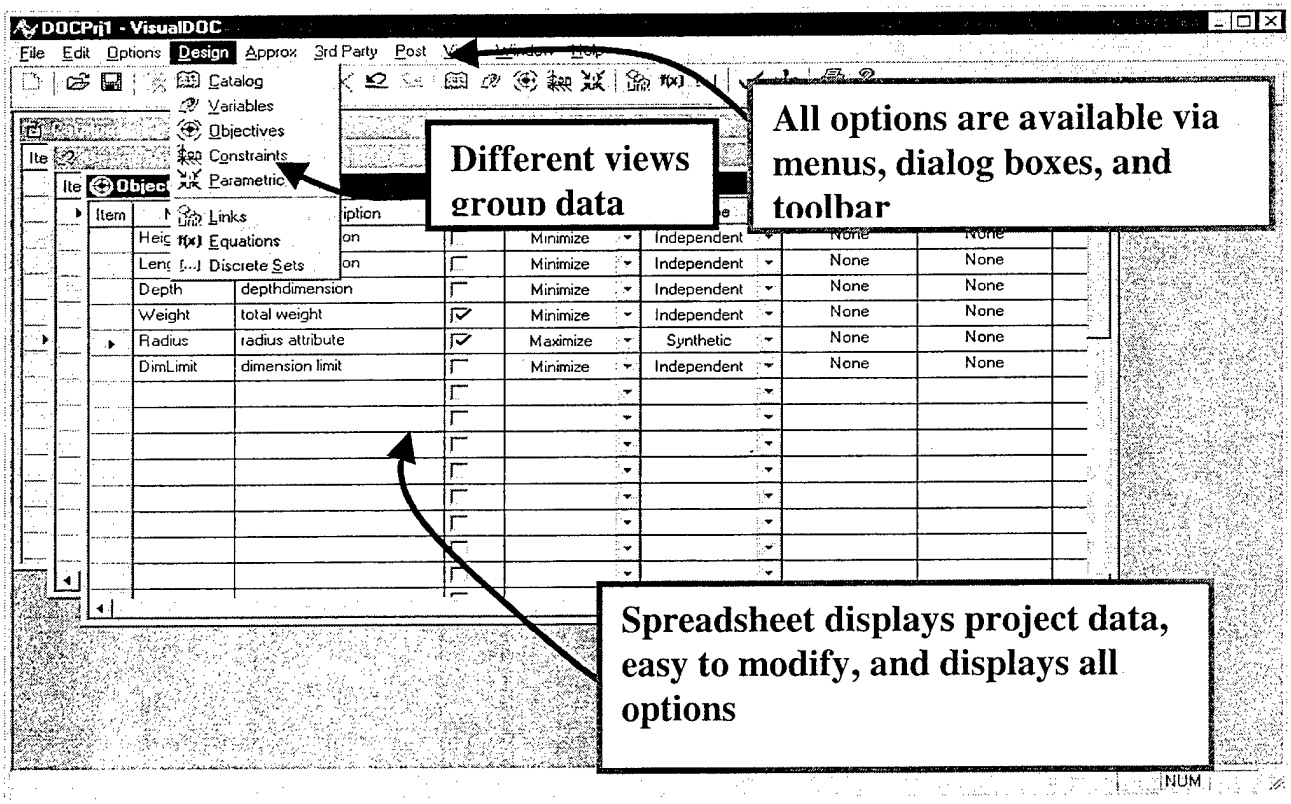

Figure 3: VisualDOC Organization on Windows 9x/NT Platforms

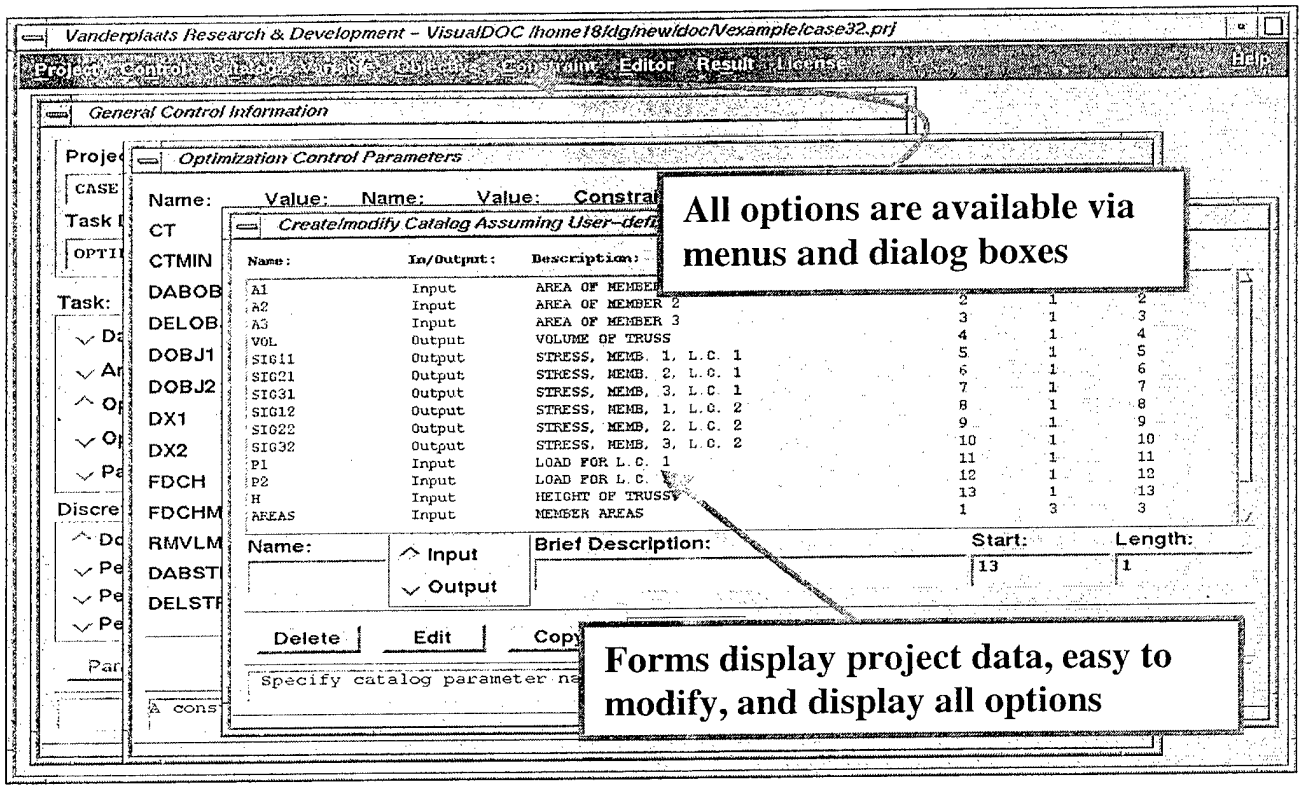

Figure 4: VisualDOC Organization on UNIX Platforms

2.4 Object Relational Database System

Figure 5 shows the basic organization of the database. It serves as a data repository for both the user interface and design modules. The database consists of three primary components: Interface Objects, Design Task Objects, and Results Objects. The arrows in figure 5 define the flow of control into and out of the database.

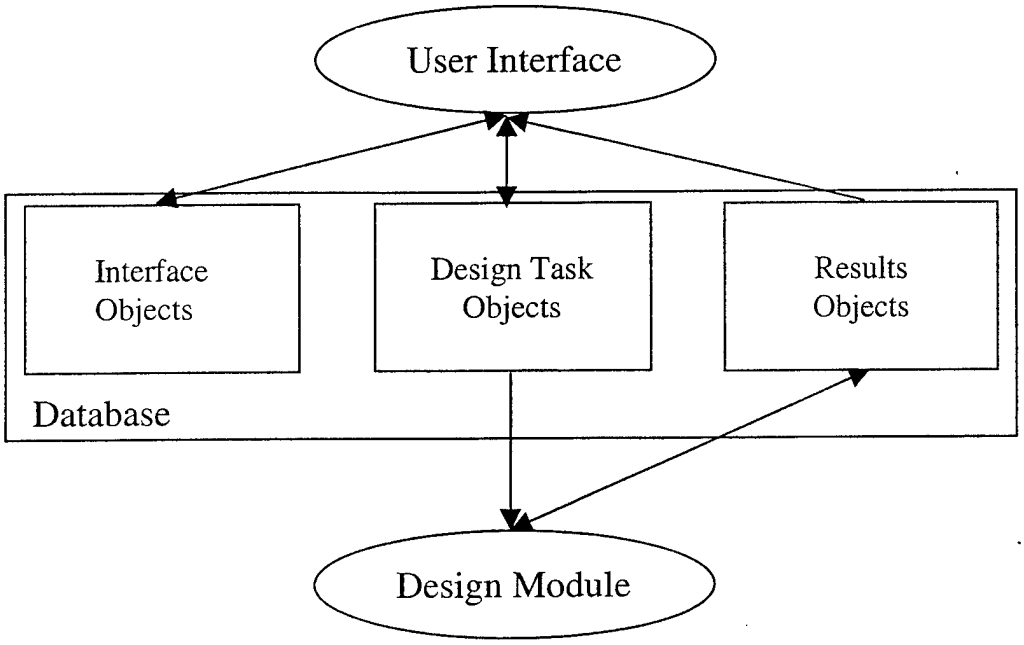

Figure 5: Database Organization

In general, objects are defined in the database for ease of use and robustness. The three primary components are described next in terms of their purposes and how they are used.

Interface Objects

Interface objects record the current state of the design project. These include current definitions of design variables, constraints, and objectives. The user interface saves a user's design project in these objects.

Design Task Objects

Design task objects are essentially a "snap-shot" of a design project for the purpose of initializing a design run. A design run utilizes one or more design modules to solve all or part of a design problem. In exploring a design problem, it is anticipated that one or more designers will create many design tasks. Design task objects define the initial starting point for the task, which includes the design method (e.g., gradient-based optimization,

DOE, etc.), variables (initial values and side constraints), constraints, objectives, and so on. Design task objects allow the database to store all the information necessary that defines the starting point for a specific design problem. Users may recall this information back into the user interface and use it to generate a new design task, or for review.

Result Objects

As a design module solves a task, it places both its intermediate and final results into the database as result objects. For example, the gradient-based optimization design module saves design variable values at each one-dimensional search point and major iteration. Likewise, response values and their gradients are saved. Each design module defines appropriate result objects for storage.

The user interface uses result objects primarily for post-processing; however, result objects may also be used for restarting and with response surface approximations.

Platform Independent Data Types

The platform independent data types for integers and floating point values are independent of the operating system and hardware platform. They overcome the big and little indian byte ordering problems encountered when writing integer and floating point values to a common database accessed by different types of computers.

Independent data types are defined for signed 4-byte integers, unsigned 4-byte integers, and 8-byte floating point values. These data types represent data internal to the database through the casting operators. They are converted into the corresponding native data types for use. Thus, the user interface and design modules use their native data types for computations.

Multi-user Capabilities

Early in the design process of the database, multi-user capabilities were identified as being particularly important because it would be likely that a user would be running the user interface while one or more design modules would be running. In this case, both the user interface and design modules would need access to the database.

Multi-user capabilities have been added through the use of operating system file locking features. File locking can lock all or part of a file. Studies have shown, however, that locking the entire file is faster for up to 8 concurrent processes rather than locking parts of a file. If more than 8 processes are accessing a file, then "record" locking gradually becomes more efficient.

It is unlikely that more than 8 processes will be concurrently accessing the database; therefore, when a module needs to access the database, the entire file is locked. Generally, the file will be locked on the order of milliseconds, so concurrent access will not likely have a significant influence on runtimes.

The database communicates with the calling processes indicating if a database transaction was successful, failed, or the database is currently locked. In the case of the database being locked, the database will first wait for a short time and try again to access the locked file. If the database is still locked, the calling process is notified and can implement additional wait logic and/or fail-safe logic.

2.5 Functional Modules

During Phase I and Phase II of the project VisualDOC became a well-developed software system. What previously was a relatively simple text interface to the optimizer

(DOC/DOT), now has evolved into a system with several functional modules, a graphical-user interface and the database. VisualDOC has closely coupled interfaces to MATLAB, LS-DYNA, and ABAQUS. The core functional module in VisualDOC is the direct continuous optimizer DOT. It is one of the best (if not the best) commercially available optimizers at the present time. Because direct continuous optimization is being performed, using VisualDOC is very efficient and robust. In addition to direct continuous optimization VisualDOC has the ability to perform optimization with discrete variables, optimization using response surface models, and Design of Experiments (DOE) studies.

2.5.1 Gradient Based Functional Module

Although the VisualDOC core functional module (DOT) is formally the same as the one of DOC/DOT, many enhancements have also been made both to DOT itself and to the modules that employ it. Many internal limitations and ambiguities have been removed.

The new version of DOT (version 5.0) has significantly more robust sequential quadratic programming algorithm than the previous version. If previously the modified method of feasible directions was the preferred method in the vast majority of problems solved, now sequential quadratic programming has became in many cases as efficient (and sometime more efficient) as the modified method of feasible directions.

VisualDOC now generates additional information for the graphical user interface and post-processing. Instead of just looking at the output file, VisualDOC allows one to graphically and in real time monitor the progress of the optimization process: how the objective function, design variables, and maximum constraint violation change from

iteration to iteration. The details about the graphical user interface and post-processing are provided in Section 2.7 of this report.

The response surface approximation module has been enhanced and improved. Particularly, the convergence criteria have been significantly modified, as well as the scaling procedure. Now optimization based on response surface approximations performs significantly better than before.

The new license checking strategy has been implemented. It does not require a lot of extra coding in the functional modules and the required code is the same for all platforms (UNIX workstations, PCs). Several levels of licensing have been implemented. The lowest level, the demo version, has very limited capabilities, whereas the highest level provides the user with all the power of VisualDOC.

Summarizing the changes in the gradient based optimization module, the gradient-based optimization in VisualDOC has become more efficient, robust, and flexible.

2.5.2 Design of Experiments (DOE) Functional Module

The Design of Experiments (DOE) functional module has been developed, and is now an integral part of VisualDOC. The main task of DOE is to help create accurate approximations of the true responses and analyze the quality of these approximations. It is accomplished by carefully selecting the points in the design space where responses should be evaluated and then fitting the curve (response surface model) to these points. Because the points in the design space are selected not at random, but in an orderly fashion, the computational expenses are minimized. Due to the use of several points in the design space that are spread out over the range of design variables, DOE creates

response surface models that tend to be a mid-range type of approximations, as opposed to local approximations generated using Taylor series expansions. Using the created response surface model, it is possible to estimate how the design variables and their combinations influence the responses. Particularly, one can determine such a combination of design variables that results in the most favorable values of responses. Another way to utilize the created response surface models is to estimate which design variables have the most influence on the responses and which design variables can be disregarded in the studies. It is also possible to use the response surface model in the formal optimization instead of calling the exact analysis. In the case when an exact analysis takes a long time, such an option provides significant computational benefits.

Currently the DOE functional module is able to create the following types of experimental designs:

- Factorial
- Composite
- Box-Behnken
- Plackett-Burman
- Notz
- Koshal
- Rechtschaffner
- Hybrid
- 3-level Taguchi orthogonal arrays

For a detailed discussion of the types of experimental designs, see the VisualDOC Supplement to the VisualDOC Reference Manual (Ref. 9).

It should be noted that the user may specify any fraction of the full factorial design, and the DOE functional module will determine the fractional factorial design of the best theoretically possible resolution. The DOE functional module is also able to import a *user-supplied design* and operate with it. Particularly, the DOE functional module can interpret candidate values of the design variables in VisualDOC as experimental design and use it in all calculations. Once the experimental design is selected, the user may slightly modify it from within VisualDOC by providing extra central and/or axial points to an already existing design.

The DOE functional module is able to analyze the geometrical characteristics of the design and create polynomial response surface approximations for the responses that were selected by the user. The approximations are created using the method of least squares. The following types of polynomials are currently supported: linear, linear with interactions, quadratic without interactions, and full quadratic. Several approximations can be created during one DOE run. The user may specify different types of approximations for different responses. After the response surface approximations have been constructed, the DOE functional module is able to perform a residual analysis for all of them. As a result of the residual analysis the user will be able to determine if the response surface model fits the experimental design well enough, and what points of the experimental design could potentially cause problems for the response surface model. Analysis of variance (ANOVA) tables for the whole response surface model and for

individual coefficients of the model are also constructed. These tables provide information about errors in the response surface model as a whole, along with the errors in determining particular coefficients of the response surface models.

Robust default values for all the input parameters of the DOE functional module have been selected and tested. Now the user does not have to worry about specifying many parameters. It is necessary to specify just the particular responses for which responses surface modules will be created. However, the experienced user still has a great deal of control over the calculations done in the DOE functional module, as well as over the amount of output information that can be obtained.

Along with the development of the DOE functional module itself, development of the interface to VisualDOC has been done. The DOE functional module is an integral part of VisualDOC 1.2. In VisualDOC 1.2, it is possible to create all of the necessary input information for the DOE functional module using the VisualDOC Interface. Both the UNIX and PC versions of the Interfaces contain all the necessary capabilities. The most useful of these capabilities are the ones that allow interaction between the DOE methods and formal optimization procedures.

VisualDOC, from the very beginning had the capability of providing candidate values of the design variables that could be used in the response surface based optimization. Now these candidate values can be directly imported into the DOE functional module using just one "click of a button" on one of the menus. Once the candidate values are inside the DOE, they are treated just like regular design of experiments and all corresponding characteristics can be obtained for these candidate values.

On the other hand, there are two ways to directly utilize the DOE results in the formal optimization procedure. The first one is to directly import the response surface approximation models into VisualDOC as synthetic equations, and then use these equations in the formal optimization instead of using expensive analysis. The importing procedure is accomplished by selecting appropriate items on the menu.

Another way of making use of the DOE results is to import the points of the design of experiment into VisualDOC as the candidate values. These candidate values can be later used in the VisualDOC optimization procedure when the "Optimization with Response Surfaces" option is selected.

One more important part of the interface of the DOE functional module with VisualDOC is post processing. It is very important to give the user a visual and simple way to interpret the results. The post processing part of the VisualDOC Interface accomplishes this. The post processing is available for both PC and UNIX Interfaces. On the PC the post processing is accomplished through Visual C++, on the UNIX – through JAVA. Currently, the following plots related to DOE functional module are provided for the user:

- Plot of the coefficients of the response surface model along with the errors for each of the coefficients.
- Plot of the responses predicted by the response surface model vs. the actual responses provided to construct the response surface model
- Plot of all the calculated residual characteristics for the particular response surface model: regular residual, standardized residual, PRESS residual, diagonal elements

of the H-hat matrix, studentized residual, R-studentized residual, Cook's *D*-statistic.

Details about the look and feel of the plots are provided in Section 2.7 of this report.

2.6 Interfaces to Third Party Analysis Software

An important aspect of VisualDOC is the ability to easily integrate VisualDOC with third party analysis programs, like ABAQUS and LS-DYNA. Such an interface is composed of three parts that (1) allows VisualDOC to change the input data to the third party analysis program, (2) executes the analysis program with the new input data and (3) extracts the desired results from the third party analysis program's output.

In version 1.0 of VisualDOC, we provided VisualDOC users with some basic functionality to integrate VisualDOC with ABAQUS and LS-DYNA. However, the original implementation was limited in its generality and usefulness, and a much improved third party interface was developed for version 2.0 of VisualDOC.

The new interface makes use of a template language to abstract the input and output data associated with a specific third party analysis program. The template language makes the interface very general so that it can be extended to almost any third party analysis program that reads input data from and writes output data to ASCII data files. Apart from being general, the new interface is also very robust. The robustness is mainly due to the abstraction of the input/output data making the interface independent of the actual position (line number) of the input/output data in the data files of the third party program. The new interface allows the user to interact with third party programs without the need to write or compile any source code.

Although the new interface is general in nature so that it could be applied to virtually any third party analysis program, we did develop a derivative specifically for ABAQUS. The ABAQUS version of the interface extracts results from the binary ABAQUS database file that contains double precision numbers as compared to the single precision numbers contained in the ABAQUS ASCII output file. The ABAQUS specific version of the interface is a direct result of interest from several customers that make use of ABAQUS.

The new third party interface has not been officially released. However, it has been used in-house on a number of GENESIS and ABAQUS example problems, and we are in the process of distributing the code to several BETA users. The new interface will be an integral part of VisualDOC version 2.0, which will allow users to setup and run a VisualDOC optimization linked to a third party analysis program from within the VisualDOC GUI.

Apart from the general interface to third party analysis programs, we also developed an interface between VisualDOC and MATLAB. The VisualDOC-MATLAB Interface is an extension to VisualDOC, aimed at MATLAB users. The VisualDOC-MATLAB Interface gives VisualDOC users access to the MATLAB computational engine for evaluating objective and constraint functions (also known as response functions) as required by the VisualDOC optimization library during an optimization run. The VisualDOC-MATLAB Interface gives the VisualDOC user full access to all the build-in MATLAB functionality through the use of MATLAB M function files. The VisualDOC-MATLAB Interface has the additional advantage that an optimization may be performed without the use of a

compiler, since VisualDOC directly calls the MATLAB engine to run the user supplied M function files.

We have a number of users that are actively using the VisualDOC-MATLAB Interface. To date all comments from our users have been very favorable. The VisualDOC-MATLAB Interface has been distributed with all copies of VisualDOC since version 1.1. This interface is especially popular among academic users and research institutions. The optimization capabilities provided through the VisualDOC-MATLAB Interface is considerably more robust than the MATLAB optimization toolkit since it offers additional features such as discrete variable optimization, response surface optimization, synthetic and linked variables, DOE etc.

2.7 Post Processing

VisualDOC contains a powerful post-processing module that is tightly integrated with the GUI. The post-processing module consists of the post-processing of design optimization results, including the objective function, design variable and maximum constraint violation history. For the Unix version of the VisualDOC GUI, we developed the graphical post-processing module using JAVA. The reason for using JAVA was to obtain similar graphical post-processing capabilities as that of the PC version of the VisualDOC GUI.

The graphical post-processing module was developed to post-process both VisualDOC optimization output data as well as output from the newly added DOE module. The module allows the user to zoom in and out of certain areas on the graphs and allows the

user to save each graph as a JPEG file. The user may then use these JPEG files, for example, to include as graphs in a report. The graphical post-processing module provides the user with the graphs described in the following sub-sections:

VisualDOC Optimization Output Data

This section of the graphical post-processing module concentrates on post-processing data related to a VisualDOC optimization run. Users create graphs by loading and processing the VisualDOC history file. The following list describes the possible graphs:

Objective Function History -- This is an X-Y line chart that shows the objective function value versus the design cycles of the optimization (see figure below).

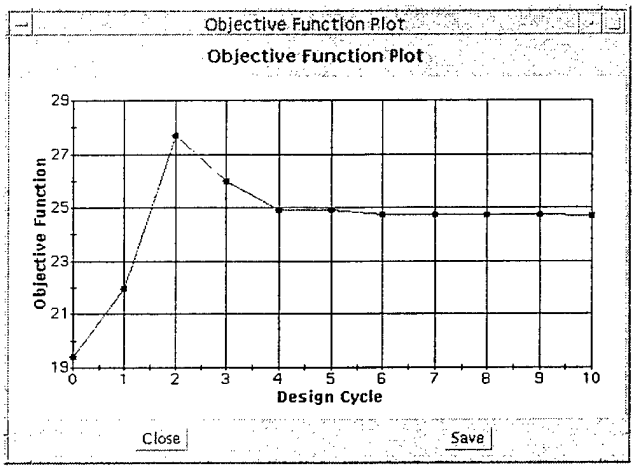

Figure 6 : Objective Function History

Maximum Constraint Violation History -- This is an X-Y line chart that shows the maximum constraint violation versus the design cycles of the optimization.

Design Variable History -- This is an X-Y line chart that shows the design variable values versus the design cycles of the optimization.

DOE Output Data

This section of the graphical post-processing module concentrates on post-process data that is related to the newly added DOE module. The graphs are created by loading and processing the DOE database file and are useful to determine the quality of the approximations created by the DOE module.

Response Surface Coefficients Plot -- This is a candle stick plot that shows the value and the standard error associated with each coefficient of the response surface approximation (see figure below). Note that all standard errors that are considered as being unreasonably large are indicated by a red bar.

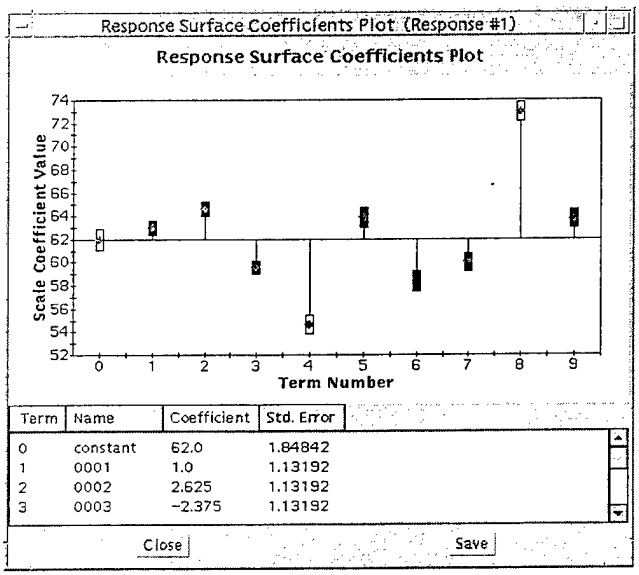

Figure 7 : Response Surface Coefficients Plot

Predicted vs. Actual Response Values -- This is an X-Y scatter plot that shows the predicted versus the actual response values.

Residuals -- This is a bar chart that shows the residual associated with each data point used to construct the approximation.

PRESS Residuals -- This is a bar chart that shows the PRESS residual associated with each data point used to construct the approximation.

H-Hat Diagonal Terms -- This is a bar chart hat that shows the H-Hat diagonal term associated with each data point used to construct the approximation. This chart includes a horizontal cut-off line that indicates critical values for the H-Hat diagonal terms (see figure below). H-Hat diagonal terms with values larger than the horizontal cut-off line are considered as being critical.

Standardized Residuals -- This is a bar chart that shows the Standardized residual associated with each data point used to construct the approximation. This chart contains horizontal cut-off lines that are useful for identifying critical values (see Figure 8).

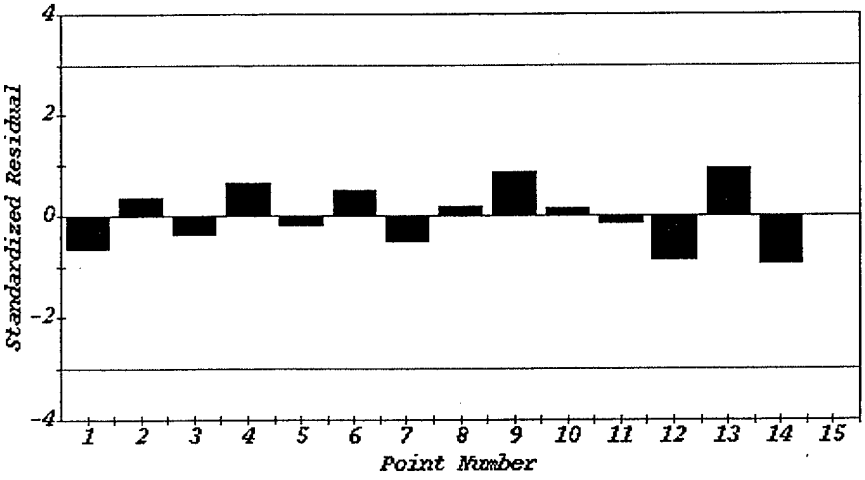

Figure 8 : Standardized Residuals Plot

Studentized Residuals -- This is a bar chart that shows the Studentized residual associated with each data point used to construct the approximation. This chart contains horizontal cut-off lines that are useful for identifying critical values.

R-Studentized Residuals -- This is a bar chart that shows the R-Studentized residual associate with each data point used to construct the approximation. This chart also contains horizontal cut-off lines that are useful for identifying critical values.

Cook's D-Statistic -- This is a bar chart that shows the Cook's D Statistic for each data point used to construct the approximation. This chart contains horizontal cut-off lines that are useful for identifying critical values.

2.8 License Manager

Since version 1.0 of VisualDOC, we have been using a strict licensing scheme that ties our software to a specific machine, referred to as a node-locked license. The user has to install the software and complete a license request containing machine specific information. VR&D then processes the license request and creates an authorization file that is sent back to the user. The final step is for the user to register the software using the authorization file received from VR&D. This process is generally completed using e-mail communication that our users find very convenient. Until now, we have been licensing all of our products on a node-locked basis only. However, since this ties a product to a specific machine, it becomes a cumbersome process for larger institutions that have several users using different machines on the same network. These institutions generally prefer a network based licensing scheme that allows any user on the network to use the licensed product. Another disadvantage of using a node locked licensing scheme, is that a

node locked license does not restrict the number of simultaneous users of a product, as long as all users are using the same machine. This becomes a problem on powerful server machines where a single license can allow a large number of users to use a licensed product at the same time. Using floating licenses, the products are no longer restricted to a single machine, but the number of users that may simultaneously use the product is restricted. Another important advantage to a floating license scheme is that the administration and distribution of the licenses becomes much easier.

We have developed a network license server program that provides floating licenses and will be distributed with version 2.0 of VisualDOC. The network license server is currently in testing at our site as well as at several BETA sites. The license server runs on a single machine on the network (either a Windows NT or UNIX workstation) and allows other machines (Windows 95/98/NT and UNIX workstations) on the same network to run a restricted number of simultaneous copies of the product. Only one physical copy of the license is required for the server machine, while all clients only require the server's IP address and a port number for connecting to the server. This significantly reduces the installation process for larger institutions. We also developed appropriate management utilities for controlling the license server. These managing utilities will be an integral part of the VisualDOC version 2.0 GUI.

3.0 Continued Research & Development

Version 2.0 of VisualDOC has several major enhancements over version 1.x, resulting in major changes in the GUI. Instead of maintaining both the PC and UNIX graphical user interfaces of VisualDOC separately, we decided to write a new interface using the JAVA

programming language. The advantage of using JAVA for the GUI is that the code is portable between UNIX and PC machines, and we, thus only need to maintain a single set of source code. Although the initial investment is high, we expect that the eventual pay off of maintaining a single set of source code will be well worth while.

We have already completed a significant amount of work on the new JAVA-based GUI and expect to release an ALPHA version within the next month and a BETA version shortly thereafter. So far, we found JAVA more than capable of providing the required GUI needs of VisualDOC. Additionally, initial test shows the code to be 100% portable and our biggest concern regarding performance issues, seems to be unfounded. Although the code is slower than a native implementation, it seems to be adequate for our GUI purposes.

4.0 Related Works

VR&D has received two research grants from NASA, LaRC to perform R&D activities in design optimization technology. These research and development activities have greatly complemented this SBIR research effort. These are briefly described below:

A SBIR Phase I project titled *Very Large Scale Structural Optimization* has recently been completed. The Phase I objective of this project was to develop "MODERN" optimizer(s) to solve problems involving a very large number of design variables and constraints, typically encountered in structural design. Initial efforts resulted in a software called BIGDOT which can handle a larger number of design variables and constraints. This optimizer has already been incorporated in the GENESIS Structural Analysis and Design Optimization program.

Another project titled *Optimization on Massively Parallel Computers* has been funded by NASA, LaRC. This project has a more immediate impact on VisualDOC. A brief overview is provided here.

Parallel/Distributed Processing

We investigated the parallelization of current optimization algorithms within VisualDOC. We developed an in-house parallel version of VisualDOC version 1.2 that we used as a test bed for investigating parallel issues. So far, we have parallelized, and tested, all finite difference gradient calculations for all VisualDOC's optimization algorithms as well as all analyses performed by the DOE module. An important aspect of the parallel version of VisualDOC is that it was designed to work on a heterogeneous network of computers. We have successfully tested our implementation on a combination of SUN, SGI, HP, IBM and Windows NT workstations. For this reason, we implemented a dynamically load-balanced parallel scheme where each processor receives a new task upon completion of its previous task until no more tasks are available. In other words, faster processors will be allocated more tasks. Also, the parallel scheme requires only minimal inter-processor communication. The design variable values are sent to the slaves, and the response values are sent back to the master.

Initial results obtained from test cases are very promising and we found significant speed up factors when performing the optimization in parallel. Note that we have only parallelized the finite difference gradient calculations and not the one-dimensional search calculations. However, tests show that when using SLP or SQP or when dealing with a

large number of design variables, the number of one-dimensional search calculations are small in comparison to the number of finite difference calculations.

We have a number of customers that are interested in this capability and plan to ship BETA versions to selected customers in the near future. We plan to make this parallel capability an integral part of VisualDOC version 2.0, allowing the user to easily define a network of parallel computers from within the GUI.

5.0 Technical Reporting

A number of technical papers related to these SBIR research efforts were published and presented in technical conferences/meetings. These are briefly outlined here:

- Dr. Dipankar Ghosh presented a paper titled *Development of a Design Optimization Interface to ABAQUS, (Ref. 4)* at the Regional ABAQUS Users' Meeting in Los Angeles, September 25, 1997. This presentation was based on the research work performed during this SBIR project. During his presentation, he received positive responses from the ABAQUS users.
- Dr. Gary Vanderplaats presented a paper titled *Development Of A Flexible Design Optimization Study Tool (Ref. 5)* at the 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, St. Louis, MO, Sept. 2-4, 1998. In this paper, we discussed the development of the design study tool (now called VisualDOC).
- Dr. Dipankar Ghosh presented a paper titled *Development of a Design Optimization Interface to LSDYNA3D (Ref. 6)* at the 5th International LS-DYNA Users Conference held on September 21-22, 1998, in Southfield, Michigan. In this paper, we showed

how VisualDOC could be used to solve problems when the LS-DYNA program is used to calculate the design responses.

- VR&D made a formal presentation of VisualDOC at the AUTOFACT 98 Conference and Exposition held in Detroit, September 29 –October 1.

- Dr. Vladimir Balabanov presented a paper titled *Multidisciplinary Optimization of a Transport Aircraft Wing (Ref. 2)* at the 40th AIAA/ASME/ASCE/AHS/ ASC Structures, Structural Dynamics and Materials conference, April, 1999, St. Louis, MO. In this paper, we showed how VisualDOC could be used to perform multidisciplinary optimization. A similar paper was also presented at the Optimization in Industry conference in Banff, Canada, June 6-11,1999. Participating in the Optimization in Industry conference gave us the opportunity to become better acquainted with the needs of particular companies and correspondingly adjust our products.

In addition to presenting technical papers at the conferences, Dr. Venkayya visited VR&D during January 15-16, 1998. During his visit, we discussed different aspects of the project. Initially, he was presented with an overview of the project and the focus of our efforts. Later, we demonstrated the status of the overall program.

6.0 Success Stories

Initially, VisualDOC Version 1.0 was released in September 1998. It provided us an opportunity to test the market, and gather user input for the next release of VisualDOC. A number of users, both from industries and research institutions, immediately started

evaluating VisualDOC. Here we would like mention two specific cases where VisualDOC is currently being used.

6.1 National Renewable Energy Laboratory (NREL)

NREL, under DOE contract, is presently developing software for a hybrid vehicle design. They have a program called "ADVISOR" for vehicle design. Recently VR&D coupled ADVISOR and VisualDOC and optimized the control system for a hybrid SUV in urban driving. This achieved nearly a 12% mileage (from 53 to 59) improvement over NREL's BEST design.

VR&D is presently negotiating with NREL and Parametric Technology Corporation (PTC) to greatly enhance and promote optimization technology for hybrid vehicles.

6.2 Visteon Automotive Systems:

Visteon Automotive Systems, Climate Control Division is presently using VisualDOC to perform optimization in CFD problems in conjunction with Pro Engineer and Fluent software. This development was initiated very recently, and we do not have any published results.

7.0 References

1. DOT Users Manual, Vanderplaats Research & Development, Inc., Colorado Springs, CO, 1999.
2. Garcelon, John, Balabanov, Vladimir and Sobiesky, Jaroslaw, "Multidisciplinary Optimization of a Transport Aircraft Wing using VisualDOC", presented at the 40th

AIAA/ASME/ASCE/AHS/ ASC Structures, Structural Dynamics and Materials Conference, April 1999, St. Louis, MO.

3. Garcelon, John, Balabanov, Vladimir and Sobiesky, Jaroslaw, "Integrating VisualDOC and GENESIS for Multidisciplinary Optimization of a Transport Aircraft Wing", presented at 3rd World Congress on Structural and Multidisciplinary Optimization, Buffalo, New York, 18-21 May, 1999.
4. Ghosh, Dipankar , and Vanderplaats, Garret., "Development of a Design Optimization Interface to ABAQUS", presented at the ABAQUS Western Users Regional Conference, Hilton and Towers Airport Hotel, Los Angeles, California, September 25, 1997.
5. Ghosh, Dipankar and Vanderplaats, Garret, "Development of a Design Optimization Interface to LSDYNA3D", presented at the 5th International LS-DYNA Users Conference on September 21-22, 1998, Southfield, Michigan.
6. Ghosh, Dipankar, Garcelon, John, Balabanov, Vladimir, Vanderplaats, Garret, "Development Of A Flexible Design Optimization Study Tool", presented at the 7th AIAA/USAF/NASA/ ISSMO Symposium on Multidisciplinary Analysis and Optimization, St. Louis, MO, Sept. 2-4, 1998.
7. Stevens, Al, C++ Database Development, MIS: Press, 1994, New York, New York.
8. VisualDOC Reference Manual, Vanderplaats Research & Development, Inc., Colorado Springs, CO, 1998.
9. VisualDOC Supplement, Vanderplaats Research & Development, Inc., Colorado Springs, CO, 1999.